Bug 1 Report Example:

Title: JMPCLD-Feature 1.1 > When a Post request is sent to the Password hash application, a job identifier is not returned immediately

Description: Given a user has the password hash application set up, when a post request is sent with a password, a job identifier is not returned immediately as expected. It is returned after about 5 seconds

Environment: <user’s test environment>

Steps to Reproduce:

Pre-conditions:   
 Port env variable is set

The user can successfully launch the application

1. Navigate to the Terminal as a valid user with access to the password hash application
2. Send a post request to the /hash endpoint

Ex: aartis-mbp-2:bin aartidhupar$ curl -X POST -H "application/json" -d '{"password":"angrymonkey"}' http://127.0.0.1:8088/hash

1. Note the response time for the job identifier to be returned

Actual Result: The job identifier is returned after about 5 seconds

Expected Result: The job identifier is returned immediately without any delay

Linked Issue: JMPCLD-Feature 1.1

Assigned to: <Dev or PO>

Priority: Low

Severity: Minor

Sprint: Spring Release Sprint 1

Fix Versions: Spring Release version 1.1

Additional bugs to be reported:

Bug - 2

The response for the post request does not confirm that the hash algorithm used to compute the password was SHA512

Bug – 3

The average time for the number of milliseconds for a single hash request should be close to 5000 ms but is close to 396967 which is 396 seconds much above the expected requirement

Assumptions/Questions?

1. The job identifiers are unique only across different sessions. Is that expected? Or we expect them to be unique across sessions?
2. Do we expect to be able to request a get with the job identifier if a session has been shut down? Or how do we expect to get back the particular password hash details if a session has been shut down and we start a new session?
3. Is there a time that the session would expire at without being specifically shut down?
4. Checking multiple connections is difficult manually. We might need a tool to be able to simulate multiple connections at the exact same time. Also need to performance test how many connections it can successfully handle at the same time successfully.

**Test Strategy:**

I have designed test cases in the **attached spreadsheet** for Functional, exploratory, positive and negative tests. Typically we also perform usability verification. However, those are more applicable for end user GUI tests.

Also, performance tests are required to test this application thoroughly for any issues.

For example, creating load on the application to see how it handles a huge number of simultaneous requests.

For example, stressing the environment incrementally to monitor the break point/recovery from the failure etc.

**Automation:**

I have basic knowledge/understanding of Automation as I have not had the chance to spend a lot of time in that area. I was only involved mostly in updating / executing existing tests and have developed a few very basic scripts by myself. There is immense opportunity to automate these scenarios. I would use any programming Java (Selenium/Cypress) or python with cucumber framework (BDD) – Given, When, Then scenario outlines.

Creating re-usable methods to script and execute the tests.

I would add the tests to kick off through Continuous Integration maybe using Jenkins where the tests are kicked off after a build is available in automated fashion.